Image Features

Imaging processes imply that ...

• Image of a scene strongly depends on pose & lighting
• Grey levels by themselves are almost meaningless
• The image itself is meaningful (where & what) but extracting the explicit meaning is complex
• Detecting similar scenes by comparing grey levels images often fails

Suggestion:
• Find **Image features** - meaningful image regions
• Use them later for description, comparison, matching

(Visual events)
Image features: Meaningful types of image regions

Desirable properties
- Tell something about the world
- Invariant, Repeatable detection
  “Invariance” – insensitivity to non-interesting variables (lighting, pose, face expression,...)
- Reduce image complexity\ dimension (?)
- Easy to compute

(General) Image Features

Point features
- Color, color invariants

Local features
- Edges
- Corners, interest points

Non-local features
- Lines, circles, ellipses, converging lines
- Texture
Edges

Edge - Significant discontinuity of intensity over a curve

Significant - large difference between grey levels
- occurs over non-zero length smooth curve

Why edges?
- Exist and common in (almost) all images
- Give information of objects in the scene
- Not so sensitive to illumination
- HVS early processing is based on edge analysis.

Many types of intensity Edges

- Object geometry,
- Occlusion,
- Albedo,
- Shadow

Also (not now)
- Color edges
- Texture edges
- Motion edges

(board: How edges are generated? Edge profiles).
Edges are informative

• The HVS perceive the edges and infers the rest
• Line drawings of object boundaries work for us. Why?

Detecting edges - Introduction

• The task: at every pixel, decide if it is an edge or not.
• Edge = location, direction, strength, (profile)

Edge - Significant discontinuity of intensity over a curve

• Goal: detecting the meaningful edges
  - Not well specified ...
  - Is “meaningful” = boundaries of object ?
  - What is “an object”
  - Detect what people prefer

• Edgel - a term for edge pixel or short curve of edge pix.
Signal and Noise

A common way to analyze processes/algorithms:
Assumption: the given data = clean “signal” + “noise”

Signal - a simplified model of the expected data
• Common clean image signal: piecewise smooth/constant

Noise - the part that is hard to predict/analyze
• Noise sources: camera noise (“true”), quantization, small scale changes (texture)
• Common noise model: white zero-mean (Gaussian) noise

Taking noise into account is essential!
Detecting edges - I

Signal detection approach

• Consider a fixed size region - a vector of grey levels \( X \)
• Consider an exactly known edge model : e.g. a straight edge in a given direction, with step profile between \( I_1, I_2 \)
• Consider two possibilities

\[
\begin{align*}
\text{Edge} & : \quad X(j) = S(j) + N(j) \quad j = 1, \ldots, m \\
\text{No Edge} & : \quad X(j) = N(j) \quad j = 1, \ldots, m
\end{align*}
\]

where \( N(j) \) is white, zero mean, Gaussian noise (b).

• Goal: make a decision between edge and no-edge

Detecting edges - I

• We look for a linear operation \( R = C^T X \)
• Why linear ? Simplest, fast, analyzable
• The decision is made by comparing the scalar \( R \) to a threshold.
• Design \( R \) to be large for edge, small for non-edge.
• \( R \) is proportional to \( \|C\| \). Let \( \|C\| = 1 \).
• \( R \) is a random variable distributed either according to \( N(0, \sigma^2) \) (no edge) or according to \( N(C^T S, \sigma^2) \) (edge).
• Which \( C \) is best ? Any decision will err less by making the \( C^T S \) largest, which happen for \( C = S / \|S\| \).

Why ?
Detecting edges - I

- Detection is done by applying the inner product on all candidate regions and threshold.
- Name: Matched filter, correlation, template matching
- Comes from digital communication
- Can detect any known signal S in white noise.
- Theoretically optimal, larger region -> better detection (why?)

Problems:
- Non-edge != noise (subtract average grey level first)
- Edge profiles are not always step-like
- Edge parameter (orientation, size, ..) are not known (search, solve, multi-detection, steerability)

Detecting edges - II

The Gradient approach

Motivation: Edge - Significant discontinuity
- Find derivatives/ gradients
- Test gradient against threshold / find local maximum

- A function $f(x,y)$  
  $\nabla f(x,y) = \begin{pmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{pmatrix} = (f_x, f_y)^T$
- The gradient $\nabla f = [\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}]$
Implementation by Discrete Gradients

- Simplest and most local derivatives
  \[
  \begin{pmatrix}
  I_x \\
  I_y
  \end{pmatrix}
  = \begin{pmatrix}
  \frac{\partial I}{\partial x} \\
  \frac{\partial I}{\partial y}
  \end{pmatrix}
  = \begin{pmatrix}
  I(x+1,y) - I(x,y) \\
  I(x,y+1) - I(x,y)
  \end{pmatrix}
  \]

- Symmetric derivatives
  \[
  \begin{pmatrix}
  I_x \\
  I_y
  \end{pmatrix}
  = \begin{pmatrix}
  \frac{\partial I}{\partial x} \\
  \frac{\partial I}{\partial y}
  \end{pmatrix}
  = \begin{pmatrix}
  I(x+1,y) - I(x-1,y) \\
  I(x,y+1) - I(x,y-1)
  \end{pmatrix}
  \]

- Evaluated by inner product / convolution with masks
  \[
  \begin{bmatrix}
  1 & -1 \\
  -1 & 1
  \end{bmatrix}
  \text{ or } \begin{bmatrix}
  1 & 0 & -1 \\
  0 & 1
  \end{bmatrix}
  \]

Detecting edges using Gradients

- Find Gradient components
- Find magnitude everywhere
- Threshold or find local maximum

\[
\| \nabla f(x,y) \| = \sqrt{\left( \frac{\partial f}{\partial x} \right)^2 + \left( \frac{\partial f}{\partial y} \right)^2}
\]
Gradient of noisy signals

- Noise makes gradient non-accurate
- Derivatives amplify noise!
- Proof: derivatives amplifies high frequencies which characterize noise but not piecewise smooth signal, so S/N ratio decreases. (example on board)

- In noisy/textured images we get many false edges

- Solution I: accept only strong responses (limited)
- Solution II: Use more than the minimal data
  - Fit a model
  - Smooth the image

Fitting a local plane using Least Squares

The image is approximated locally by a Taylor series - a plane
\[ I(x,y) = I(0,0) + xI_x + yI_y + ... + y + c \]

For every point in the neighborhood, with coordinates and gray level \( x_j, y_j, I_j = I(x_j, y_j) \), \( I_j \approx ax_j + by_j + c \).

Choose a neighborhood with \( N>3 \) points and get an over-determined set of equations

A least squares (LS) solution reduces estimation error
\[ (a^*, b^*, c^*) = \arg \min \sum_j \left( ax_j + by_j + c - I_j \right)^2 \]
Finding derivatives by fitting a local plane using LS

Consider a 3x3 neighborhood with gray levels

On a unit step, centered, grid,

\[ I_j \approx ax_j + by_j + c \]

Solving the LS we get

\[ \theta^* = \begin{pmatrix} a \\ b \\ c \end{pmatrix} = \left( H^T H \right)^{-1} H^T I \]

Which corresponds to the masks (Prewitt edge operators)

\[
\begin{bmatrix}
-1 & 0 & 1 \\
\cdots & \cdots & \cdots \\
1 & -1 & 1
\end{bmatrix}
\]

Solving a least Square problem

The minimization

\[ \theta^* = \arg \min C(\theta) = \arg \min \|H\theta - I\|^2_2 = \arg \min \left( H\theta - I \right)^T \left( H\theta - I \right) \]

\[
\frac{\partial C(\theta)}{\partial \theta} = \frac{\partial}{\partial \theta} \left( \theta^T H^T H\theta - 2\theta^T H^T I + I^T I \right) = 2H^T H\theta - 2H^T I = 0
\]

where

\[
\frac{\partial C(\theta)}{\partial \theta} = \begin{pmatrix} \frac{\partial C(\theta)}{\partial \theta_1} \\ \vdots \\ \frac{\partial C(\theta)}{\partial \theta_m} \end{pmatrix} = \frac{\partial}{\partial \theta} \left( \theta^T A_{sym} \theta \right) = 2A_{sym}\theta \quad \frac{\partial}{\partial \theta} \left( \theta^T a \right) = a
\]

Solving the normal equations pseudo-inverse

\[ H^T H\theta - H^T I = 0 \Rightarrow \theta^* = \left( H^T H \right)^{-1} H^T I \]
Smoothing and Deriving

- The “alternative” (to model fitting): reduce the noise by smoothing the image.

- For linear smoothing, which one should come first?
  \[ I \ast (\text{smooth}) \ast (\text{gradient}) \]
  \[ I \ast (\text{gradient}) \ast (\text{smooth}) \]

  Does not matter! Why?

- Common: \[ I \ast [ (\text{gradient}) \ast (\text{smooth}) ] \]

Combined operators

Prewitt is a combined operator
Sobel is another example

\[
\begin{bmatrix}
-1 & 0 & 1 \\
-2 & 0 & 2 \\
-1 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
1 & 2 & 1 \\
0 & 0 & 0 \\
-1 & -2 & -1
\end{bmatrix}
\]

- Note: Smoothing and differentiating are contrasting operations, but in 2D we can do both!

- Thresholding the Sobel operator smoothed gradient
- Thresholding produces thick edges

Figure 4.7: Left: output of Sobel edge enhancer run on Plane 6.1. Middle: edge-drawn.
Gaussian smoothing - the common choice

- Good properties as smoother: symmetric, emphasize center, separable, infinitely deriveable, (semi-group, causality).

\[ G(x, \sigma) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{x^2}{2\sigma^2}} \]

Gaussian smoothing

\[ I = I_{original} \ast G(x, y, \sigma) = I_{original} \ast \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \]

Smoothing reduces both noise (+) and detail (-)
Gaussian derivative

Combined Gaussian derivative
I * [ (gradient) * (smooth) ]

\[ G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \]

\[ G_x = \frac{\partial G}{\partial x} (x, y, \sigma) = -\frac{x}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \]

\[ G_y = \frac{\partial G}{\partial y} (x, y, \sigma) = -\frac{y}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \]

Board: 1D plot

Gaussian derivatives responses

1 pixel 3 pixels 7 pixels
Edge detection with Laplacian

Basic idea: Finding the maximum of 1st derivative ~ finding zero crossing of 2nd derivatives

Laplacian

$$\nabla^2 I = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}$$

Elegant (one filter) but noisy

Edge det. with Laplacian of Gaussian - LoG

$$\nabla^2 I = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}$$

Marr-Hildreth: smooth first..

$$\nabla^2 (I \ast G(x, y)) = I \ast \nabla^2 G(x, y)$$
Canny edge detector

1. Apply Gaussian smoothing
2. Calculate Gradient magnitude and direction
3. Initialize: Choose a point where GradMag > Th_high
4. Thin (Non maxima suppression): check neighbors in Grad direction and choose maximum

Two thresholds: Hysteresis
Generates more continuous curves
How much smoothing? (scale)

Small scale: more detail, noise sensitive
Coarse scale: less noisy, miss details, location inaccurate, smooth over other objects.

Using spatial edge support

- In real images, edges are not isolated
- Smooth curves are common
- Isolated edgels are often less important.

- A detected edgel enhances other collinear edgels
- Grouping/saliency approaches use such enhancement
Edge detectors - summary

- Sobel (Prewitt, Roberts,...) - old
- Laplacian - Elegant, perhaps HVS compatible
- Gaussian derivatives, Canny - commonly used
- Edison - uses gradient + a test on step edge profile

- Often, strong smooth edge is not an object boundary, and vice-versa.
- Color, texture edges, multi-scale detection
- Recent trend:
  - Learn from examples how an edge looks like.
  - Optimize performance of manually annotated DB of images